

# Fundamental Limits of Caching

Mohammad Ali Maddah-Ali and Urs Niesen

## Abstract

Caching is a technique to reduce peak traffic rates by prefetching popular content in memories at the end users. This paper proposes a novel caching approach that can achieve a significantly larger reduction in peak rate compared to previously known caching schemes. In particular, the improvement can be on the order of the number of end users in the network. Conventionally, cache memories are exploited by delivering requested contents in part locally rather than through the network. The gain offered by this approach, which we term *local caching gain*, depends on the *local* cache size (i.e., the cache available at each individual user). In this paper, we introduce and exploit a second, *global*, caching gain, which is not utilized by conventional caching schemes. This gain depends on the aggregate *global* cache size (i.e., the cumulative cache available at all users), even though there is no cooperation among the caches.

To evaluate and isolate these two gains, we introduce a new, information-theoretic formulation of the caching problem focusing on its basic structure. For this setting, the proposed scheme exploits both local and global caching gains, leading to a multiplicative improvement in the peak rate compared to previously known schemes. Moreover, we argue that the performance of the proposed scheme is within a constant factor from the information-theoretic optimum for all values of the problem parameters.

## I. INTRODUCTION

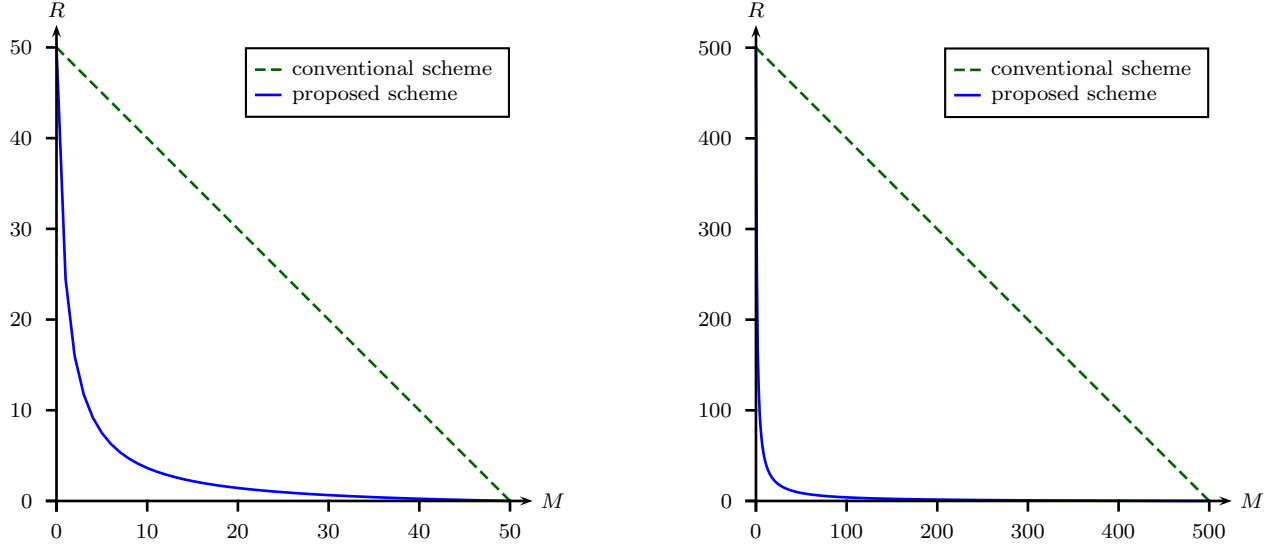
The high temporal variability of network traffic results in communication systems that are congested during peak-traffic times and underutilized during off-peak times. One approach to reduce peak traffic is to take advantage of memories distributed across the network (at end users, servers, routers, ...) to duplicate content. This duplication of content, called content placement or caching, is performed during off-peak hours when network resources are abundant. During peak hours, when network resources are scarce, user requests can then be served from these caches, reducing network congestion. In this manner, caching effectively allows to shift traffic from peak to off-peak hours, thereby smoothing out traffic variability and reducing congestion.

From the above discussion, we can see that the caching problem consists of two distinct phases. The first phase is the *placement phase*, which is based solely on the statistics of the user demands. In this phase, the network is not congested, and the main limitation is the size of the cache memories. The second phase is the *delivery phase*, which is performed once the actual demands of the users have been revealed. In this phase, the network is congested, and the main limitation is the rate required to serve the requested content.

Various versions of this problem have been studied, with the focus being mainly on exploiting the history or statistics of the user demands [1]–[7]. In these papers, the operation of the delivery phase is fixed to consist of simple orthogonal unicast or multicast transmissions. Assuming this method of delivery, the cache placement is then optimized. The gain of caching in this approach results from making popular content available locally. In another line of research, the objective is to optimize the delivery phase for fixed known cache contents and for specific demands [8], [9].

As pointed out above, the gain from traditional caching approaches derives from making content available locally: if a user requests some content that is stored in its cache, it can be served from its local memory. We hence call this the *local caching gain*. This gain is relevant if the local cache memory is large enough such that a sizable fraction of the total (popular) content can be stored locally. On the other hand, if the size of the local caches is small compared to the total amount of content, then this gain is insignificant.

In this paper, we propose a novel caching approach that, in addition to the local caching gain, is able to achieve a *global caching gain*. This gain derives from *jointly* optimizing both the caching and delivery phases, ensuring that in the delivery phase several *different* demands can be satisfied with a single multicast transmission. Since the cache placement is performed without knowledge of the actual demands, in order to achieve this gain the placement must be carefully designed such that these multicasting opportunities are created *simultaneously* for all possible requests in the delivery phase. We show that this global caching gain is relevant if the aggregate global cache size is large enough compared to the total amount of content. Thus, even though the caches cannot cooperate, the sum of the cache sizes becomes an important system parameter.



(a) Caching performance for  $N = 50$  files and  $K = 50$  users. For a memory able to store  $M = 10$  files, the rate required by the proposed scheme is a factor 11 smaller than the rate required by conventional schemes.

(b) Caching performance for  $N = 500$  files and  $K = 500$  users. For a memory able to store  $M = 100$  files, the rate required by the proposed scheme is a factor 101 smaller than the rate required by conventional schemes.

Fig. 1. Rate  $R$  required in the delivery phase as a function of memory size  $M$ . The dashed green curve shows the performance of conventional caching strategies. The solid blue curve is the performance of the proposed new caching scheme.

To formally analyze the performance of the proposed caching scheme, and in order to evaluate and isolate these two gains, we introduce a new, information-theoretic formulation of the caching problem focusing on its basic structure. In our setting,  $K$  users are connected to a server through a shared, error-free link. The server has a database of  $N$  files of equal size and equal popularity. Each of the users has access to a cache memory big enough to store  $M$  of the files. During the placement phase, the caches are filled as a function of the database. During the delivery phase, each user may ask for any one of the  $N$  possible files. The objective is to design the placement and delivery phases such that the rate in the delivery phase is minimized. For simplicity, we restrict the discussion in the introduction section to the most relevant case, in which the number of files  $N$  is larger than the number of users  $K$ . The main results are presented later for the general case.

In the above setting, the rate in the delivery phase without any caching is equal to  $K$  files. The two caching gains mentioned above are shown to be as follows:

- *Local caching gain*: The rate is reduced by a factor  $1 - M/N$ . This gain depends on the normalized *local* cache size  $M/N$ .
- *Global caching gain*: The rate is reduced by a factor  $\frac{1}{1+KM/N}$ . This gain depends on the normalized *cumulative* cache size  $KM/N$ .

Observe that the local caching gain is relevant only if the cache size  $M$  is on the order of the number of

files  $N$ . On the other hand, the global caching gain is relevant whenever the cumulative cache size  $KM$  is on the order of (or larger than) the number of files  $N$ .

As pointed out earlier, conventional caching solutions exploit only the local caching gain. In this setting, the rate in the delivery phase of conventional caching schemes is

$$K \cdot (1 - M/N).$$

On the other hand, the new caching scheme proposed in this paper attains both the local and the global caching gains. The rate in the delivery phase of the proposed scheme is

$$K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N}.$$

Moreover, by deriving fundamental lower bounds on the rate required in the delivery phase, we show that this rate is within a factor 12 of the information-theoretic optimum for all values of  $N$ ,  $K$ , and  $M$ .

To obtain some intuition about the effect of these two gains, let us compare the rates of the conventional scheme (achieving only the local gain) versus the proposed scheme (achieving both the local and global gains) for a system with  $N = 50$  files and  $K = 50$  users as shown in Fig. 1(a). When each user has a cache memory large enough to store  $M = 10$  files, the rate of the conventional scheme corresponds to sending 40 files over the shared link, while the proposed scheme achieves a rate corresponding to sending only 3.6 files: a reduction by a factor 11 in rate.

The gain of the proposed scheme compared to conventional strategies is even more pronounced for larger values of  $N$  and  $K$ , as can be seen in Fig. 1(b) for the scenario with 500 files and users. For general  $N$  and  $K$ , the performance gain of the proposed scheme over conventional caching strategies can be up to a factor of order  $\min\{N, K\}$ .

The remainder of this paper is organized as follows. Section II formally introduces our information-theoretic formulation of the caching problem. Section III presents main results, which are illustrated with examples in Section IV. Sections V–VII contain proofs.

## II. PROBLEM SETTING

Before formally introducing the problem in Section II-B, we start with an informal description in Section II-A.

### A. Informal Problem Description

We consider a system with one server connected through a shared, error-free link to  $K$  users. The server has access to a database of  $N$  files  $W_1, \dots, W_N$  each of size  $F$  bits. Each user  $k$  has an isolated cache memory  $Z_k$  of size  $MF$  bits for some real number  $M \in [0, N]$ .

The system operates in two phases: a *placement phase* and a *delivery phase*. In the placement phase, the users are given access to the entire database  $W_1, \dots, W_N$  of files. Each user  $k$  is then able to fill the content of its cache  $Z_k$  using the database. In the delivery phase, only the server has access to the database of files. Each user  $k$  requests one of the files  $W_{d_k}$  in the database. The server is informed of these requests and proceeds by transmitting a signal  $X_{(d_1, \dots, d_K)}$  of size  $RF$  bits over the shared link for some fixed real number  $R$ . Using the content  $Z_k$  of its cache and the signal  $X_{(d_1, \dots, d_K)}$  received over the shared link, each user  $k$  aims to reconstruct its requested file  $W_{d_k}$ .

We say that a memory-rate pair  $(M, R)$  is *achievable for requests*  $d_1, \dots, d_K$  if every user  $k$  is able to recover its desired file  $W_k$  (with high probability). A memory-rate pair  $(M, R)$  is said to be *achievable* if this pair is achievable for every possible request  $d_1, \dots, d_K$  in the delivery phase. Finally, we denote by  $R^*(M)$  the smallest rate  $R$  such that  $(M, R)$  is achievable. The function  $R^*(M)$  describes the *memory-rate tradeoff* for the caching problem. The aim of this paper is to characterize this memory-rate tradeoff. In other words, we aim to find the minimum rate of communication over the shared link at which all possible demand tuples can be satisfied.

We illustrate these definitions with the example of the caching strategy employed by conventional caching systems.

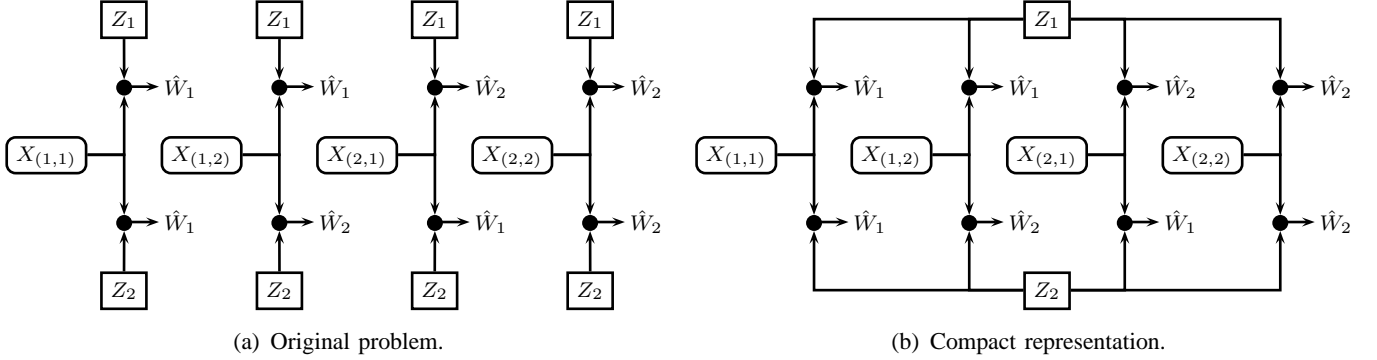


Fig. 2. The caching problem with  $N = 2$  files and  $K = 2$  users.  $Z_1$  and  $Z_2$  are caches of size  $MF$  bits.  $X_{(d_1, d_2)}$  is the signal of size  $RF$  bits sent over the shared link responding to user one and two requesting files  $W_{d_1}$  and  $W_{d_2}$ , respectively.

**Example 1 (Conventional Scheme).** Consider the caching problem with  $N = 2$  files  $W_1, W_2$  and  $K = 2$  users as depicted in Fig. 2. In the placement phase, users one and two fill the contents  $Z_1$  and  $Z_2$  of their respective caches as a function of both files  $W_1$  and  $W_2$ . In the delivery phase, each user requests a file, say user one request  $W_2$  and user two requests  $W_1$ . As a response to these requests, the server transmits  $X_{(2,1)}$  over the shared link. Using only  $Z_1$  and  $X_{(2,1)}$ , user recovers an estimate  $\hat{W}_2$  of the desired message  $W_2$ . Similarly, using only  $Z_2$  and  $X_{(2,1)}$ , user two recovers an estimate  $\hat{W}_1$  of the desired message  $W_1$ .

For concreteness, assume that the caches have size  $F$  so that  $M = 1$ . One possible caching strategy is to place an equal fraction of each of the  $N$  files in the cache. In this example, each of the two caches  $Z_1$  and  $Z_2$  would then consist of the first halves of the files  $W_1$  and  $W_2$ . In the delivery phase, the server simply transmits the second halves of the two files. This requires  $F$  bits to be sent so that  $R = 1$ . Clearly, from the content of their caches and the signal sent over the shared link, the users can recover the requested files.

We refer to the caching strategy described in the last paragraph as the *conventional caching scheme*, since it is the strategy employed by traditional caching systems. For general  $N$ ,  $K$ , and  $M$ , in the conventional scheme each user caches the same  $M/N$  fraction of each file. In the delivery phase, the remaining  $1 - M/N$  fraction of any requested file needs to be transmitted over the shared link. Therefore, the delivery rate in the conventional scheme is

$$\min\{N, K\} \cdot (1 - M/N).$$

As we shall see in the following, this conventional caching strategy can be significantly improved upon, especially for large values of  $K$  and  $N$ .  $\diamond$

### B. Formal Problem Statement

We now provide the formal definition of the information-theoretic caching problem. Let  $(W_n)_{n=1}^N$  be  $N$  independent random variables each uniformly distributed over

$$[2^F] \triangleq \{1, 2, \dots, 2^F\}$$

for some  $F \in \mathbb{N}$ . Each  $W_n$  represents a file of size  $F$  bits. A  $(M, R)$  caching scheme consists of  $K$  caching functions,  $N^K$  encoding functions, and  $KN^K$  decoding functions.

The  $K$  caching functions

$$\phi_k: [2^F]^N \rightarrow [2^{FM}]$$

map the files  $W_1, \dots, W_N$  into the cache content

$$Z_k \triangleq \phi_k(W_1, \dots, W_N)$$

for each user  $k \in [K]$  during the placement phase. The  $N^K$  encoding functions

$$\psi_{(d_1, \dots, d_K)} : [2^F]^N \rightarrow [2^{\lfloor FR \rfloor}]$$

map the files  $W_1, \dots, W_N$  to the input

$$X_{(d_1, \dots, d_K)} \triangleq \psi_{(d_1, \dots, d_K)}(W_1, \dots, W_N)$$

of the shared link responding to the requests  $(d_1, \dots, d_K) \in [N]^K$  during the delivery phase. Finally, the  $KN^K$  decoding functions

$$\mu_{(d_1, \dots, d_K), k} : [2^{\lfloor RF \rfloor}] \times [2^{\lfloor FM \rfloor}] \rightarrow [2^F]$$

map the signal received over the shared link  $X_{(d_1, \dots, d_K)}$  and the cache content  $Z_k$  to the estimate

$$\hat{W}_{(d_1, \dots, d_K), k} \triangleq \mu_{(d_1, \dots, d_K), k}(X_{(d_1, \dots, d_K)}, Z_k)$$

of the requested file  $W_{d_k}$  of user  $k \in [K]$ . The probability of error is defined as

$$\max_{(d_1, \dots, d_K) \in [N]^K} \max_{k \in [K]} \mathbb{P}(\hat{W}_{(d_1, \dots, d_K), k} \neq W_{d_k}).$$

**Definition.** The pair  $(M, R)$  is *achievable* if for every  $\varepsilon > 0$  and every large enough file size  $F$  there exists a  $(M, R)$  caching scheme with probability of error less than  $\varepsilon$ . We define the *memory-rate tradeoff*

$$R^*(M) \triangleq \inf \{ R : (M, R) \text{ is achievable} \}.$$

### III. MAIN RESULTS

For ease of exposition, we first consider the case  $N \geq K$ . The next result presents an achievable rate, yielding an upper bound on the memory-rate tradeoff  $R^*(M)$ .

**Theorem 1.A.** For  $N \in \mathbb{N}$  files and  $K \in \{1, 2, \dots, N\}$  users each with cache of size  $M \in \frac{N}{K} \cdot \{0, 1, \dots, K\}$ ,

$$R^*(M) \leq R(M) \triangleq K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N}$$

is achievable. For general  $0 \leq M \leq N$ , the lower convex envelope of these points is achievable.

The caching algorithm achieving  $R(M)$  is described in Section V. The rate  $R(M)$  in Theorem 1.A is composed of three distinct factors. Observe that in the absence of caches at the users (i.e.,  $M = 0$ ), the worst-case rate with unicast delivery is  $K$ . This is the first factor in  $R(M)$ .

Recall from Example 1 in Section II-A that the conventional caching scheme achieves a rate of

$$K \cdot (1 - M/N). \tag{1}$$

Therefore, the conventional scheme offers a caching gain of a factor  $1 - M/N$ , which we call the *local caching gain*. This local caching gain is the second factor in  $R(M)$ . Observe that this local gain is a function of the normalized *local* memory size  $M/N$  and is relevant whenever  $M$  is on the order of  $N$ .

Finally, the third factor in  $R(M)$  is the *global caching gain*  $\frac{1}{1 + KM/N}$ . This gain is a function of the normalized *global* or *cumulative* memory size  $KM/N$  and is relevant whenever  $KM$  is on the order of (or larger than)  $N$ . This global gain is to be interpreted as a simultaneous multicasting gain available for all possible demands. Note that, since the number of users is smaller than the number of files, in the worst case all users request different files. Hence, when  $N \geq K$ , there are no natural multicasting opportunities. The scheme proposed in Theorem 1.A carefully designs the cache placement in order to

create multicasting opportunities in the delivery phase even among users that request different files. Since the placement phase is performed without knowledge of the actual demands, care must be taken to ensure that the same multicasting opportunities are created simultaneously for every possible set of requests in the delivery phase.

We point out that the conventional caching scheme introduced in Example 1 achieves only the local caching gain, whereas the caching strategy proposed in Theorem 1.A achieves both the local as well as the global caching gains. The following two examples compare these two gains.

**Example 2** ( $\Theta(K)$  *Improvement in Rate*). Consider a situation with the same number of users as files, i.e.,  $N = K$ . Assume each user has enough local cache memory for half of the files so that  $M = N/2$ . Then the local caching gain is  $1/2$  and the global caching gain is  $1/(1 + K/2)$ . By (1), the conventional scheme achieves a rate of  $K/2$ . On the other hand, by Theorem 1.A, the scheme proposed in this paper achieves a rate of  $(K/2)/(1 + K/2) < 1$ : a reduction by more than a factor  $K/2$  in rate compared to the conventional scheme. We refer the reader to Fig. 1 in Section I for a visualization of the effect of this improvement.  $\diamond$

**Example 3** ( $\Theta(K)$  *Improvement in Slope*). In this example, we compare the performance of the proposed and conventional schemes for small values of the local cache size  $M$ . We consider again the case  $N = K$ . From (1), the slope of the rate of the conventional scheme as a function of cache size  $M$  around  $M = 0$  is equal to  $-1$ . On the other hand, by Theorem 1.A, the scheme proposed in this paper has a slope of less than  $-K/2$  around  $M = 0$ . Therefore, the rate of the proposed scheme reduces with the local cache size at least  $K/2$  times faster than the conventional scheme. Comparing the rates of conventional and proposed scheme in Fig. 1 in Section I for small values of  $M$  illustrates the effect of this improvement.  $\diamond$

For the case  $N < K$ , the achievable rate is stated in the following theorem.

**Theorem 1.B.** *For  $N \in \mathbb{N}$  files and  $K \in \{N + 1, N + 2, \dots\}$  users each with cache of size  $M \in \frac{N}{K} \cdot \{0, 1, \dots, K\}$ ,*

$$R^*(M) \leq R(M) \triangleq K \cdot (1 - M/N) \cdot \min \left\{ \frac{1}{1 + KM/N}, \frac{N}{K} \right\}$$

*is achievable. For general  $0 \leq M \leq N$ , the lower convex envelope of these points is achievable.*

The proof of Theorem 1.B is reported in Section V. Again, the rate  $R(M)$  in Theorem 1.B is composed of three factors. The first two factors are the same as for the case  $N \geq K$  (worst-case rate with unicast delivery  $K$ , local caching gain  $1 - M/N$ ).

However, the third factor (the global gain) is different. For  $N < K$ , this gain is the minimum of two terms. The first term  $\frac{1}{1 + KM/N}$  is the same as in Theorem 1.A for  $N \geq K$ . Recall that this global gain is a simultaneous multicasting gain created by careful cache placement to allow multicasting opportunities for all possible requests in the delivery phase. However, for a situation with fewer files than users, there exists already a natural multicasting opportunity: by multicasting all  $N$  files to the  $K$  users, we can achieve a gain of  $N/K$ . The scheme in Theorem 1.B achieves the minimum of these two gains. We point out that for  $M \geq 1 - N/K$  this minimum is achieved by the first of the two gains. In other words, the natural multicasting gain is relevant only when the memory size is very small.

Having established an upper bound on the memory-rate tradeoff  $R^*(M)$ , we proceed with a lower bound on it.

**Theorem 2.** *For  $N \in \mathbb{N}$  files and  $K \in \mathbb{N}$  users each with cache of size  $0 \leq M \leq N$ ,*

$$R^*(M) \geq \max_{s \in \{1, \dots, \min\{N, K\}\}} \left( s - \frac{s}{\lfloor N/s \rfloor} M \right).$$

The proof of Theorem 2, presented in Section VI, is based on a cut-set bound argument. While tighter lower bounds on  $R^*(M)$  can be derived using stronger arguments than the cut-set bound (see the discussion in Example 4 in Section IV), these are not required for the purposes of this paper.

Comparing the achievable rate  $R(M)$  in Theorem 1 with the lower bound in Theorem 2, we obtain the following approximation result for the memory-rate tradeoff  $R^*(M)$ .

**Theorem 3.** For  $N \in \mathbb{N}$  files and  $K \in \mathbb{N}$  users each with cache of size  $0 \leq M \leq N$ ,

$$1 \leq \frac{R(M)}{R^*(M)} \leq 12,$$

with the achievable rate  $R(M)$  as defined in Theorem 1.

The proof of Theorem 3 is presented in Section VII. The bound  $R(M)/R^*(M) \leq 12$  on the approximation ratio of the proposed caching scheme is somewhat loose due to the analytical bounding techniques that were used. Numerical simulations suggest that

$$\frac{R(M)}{R^*(M)} \leq 5$$

for all  $N$ ,  $K$ , and  $0 \leq M \leq N$ .

Theorem 3 shows that the rate  $R(M)$  of the proposed scheme in Theorem 1 is close to the information-theoretic optimum  $R^*(M)$  for all values of the system parameters. More precisely, it shows that no scheme can improve upon the rate  $R(M)$  of the proposed scheme by more than a factor 12. This also implies that the local and global caching gains identified in this paper are fundamental: there are no other significant caching gains beyond these two.

#### IV. EXAMPLES

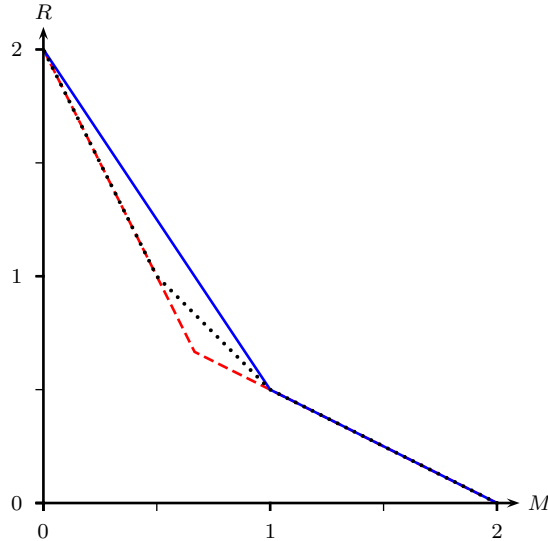


Fig. 3. Memory-rate tradeoff for  $N = 2$  files  $K = 2$  users. The achievable rate  $R(M)$  from Theorem 1 is indicated by the solid blue curve. The lower bound on  $R^*(M)$  from Theorem 2 is indicated by the dashed red curve. For the  $N = K = 2$  case,  $R^*(M)$  can be found exactly and is indicated by the dotted black curve.

**Example 4.** Consider the case  $N = K = 2$ , so that there are two files, say  $W_1 = A, W_2 = B$ , and two users each with cache memory of size  $M$ . The upper and lower bounds in Theorems 1 and 2 on the memory-rate tradeoff  $R^*(M)$  are depicted in Fig. 3. To illustrate the proof techniques, we now show how these two bounds are derived for this simple setting.

We start with the upper bound in Theorem 1. First, let us consider the two extreme cases  $M = 0$  and  $M = N$ . If  $M = 0$ , the server can always transmit both files  $A$  and  $B$  over the shared link. Since this satisfies every possible request, the  $(M, R)$  pair  $(0, 2)$  is achievable. If  $M = 2$ , each user can cache both

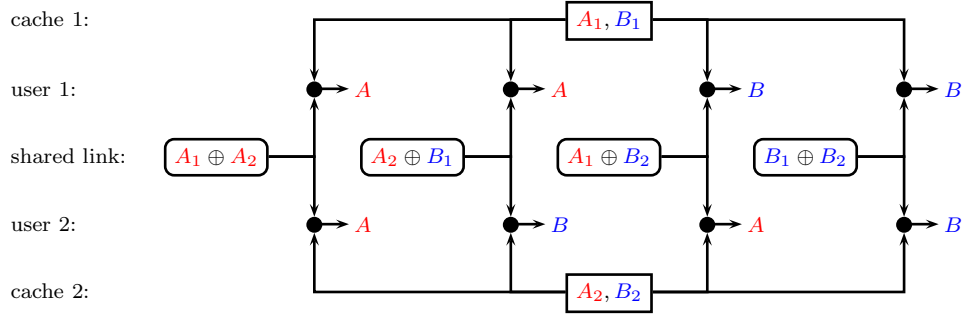


Fig. 4. Caching strategy for  $N = 2$  files and  $K = 2$  users with cache size  $M = 1$ . Each file is split into two subfiles of size  $1/2$ , i.e.,  $A = (A_1, A_2)$  and  $B = (B_1, B_2)$ . The scheme achieves rate  $R = 1/2$ .

files  $A$  and  $B$  in the placement phase. Therefore, no communication is needed in the delivery phase and the  $(M, R)$  pair  $(2, 0)$  is achievable.

Consider then a cache size  $M = 1$ . The caching scheme achieving the upper bound in Theorem 1 is as follows (see Fig. 4). We split both files  $A$  and  $B$  into two subfiles of equal size, i.e.,  $A = (A_1, A_2)$  and  $B = (B_1, B_2)$ . In the placement phase, we set  $Z_1 = (A_1, B_1)$  and  $Z_2 = (A_2, B_2)$ . In words, each user caches one exclusive part of each file. For the delivery phase, assume for example that user one requests file  $A$  and user two requests file  $B$ . Given that user one already has subfile  $A_1$  of  $A$ , it only needs to obtain the missing subfile  $A_2$ , which is cached in the second user's memory  $Z_2$ . Similarly, user two only needs to obtain the missing subfile  $B_1$ , which is cached in the first user's memory  $Z_1$ . In other words, each user has one subfile of the file that the other user needs.

The server can in this case simply transmit  $X_{(1,2)} = A_2 \oplus B_1$ , where  $\oplus$  denotes bitwise XOR. Since user one already has  $B_1$ , it can recover  $A_2$  from  $A_2 \oplus B_1$ . Similarly, since user two already has  $A_2$ , it can recover  $B_1$  from  $A_2 \oplus B_1$ . Thus, the signal  $A_2 \oplus B_1$  received over the shared link helps both users to effectively exchange the missing subfiles available in the cache of the other user.

The signals sent over the shared link for all other requests are depicted in Fig. 4. One can see that in all cases the signal is constructed using the same logic of exchanging the missing subfile. This proves the achievability of the  $(M, R)$  pair  $(1, 1/2)$ .

So far, we have shown that the  $(M, R)$  pairs  $(0, 2)$ ,  $(1, 1/2)$ , and  $(2, 0)$  are achievable. On the other hand, it is easy to see that if any two points  $(M_1, R_1)$  and  $(M_2, R_2)$  are achievable, the line connecting these two points is also achievable. This can be shown by dividing the cache memories and the transmitted signal proportionally between the two caching schemes. Together, this proves the achievability of the solid blue curve in Fig. 3, which coincides with the upper bound stated in Theorem 1.

We continue by analyzing the lower bound on  $R^*(M)$  in Theorem 2. The proof relies on the cut-set bound. We consider two cuts. Referring to Fig. 2(b) in Section II, the first cut separates  $(X_{(1,2)}, Z_1, Z_2)$  from  $(\hat{W}_1, \hat{W}_2)$ . Assume  $(M, R)$  is an achievable memory-rate pair. Then this cut has capacity at most  $RF + 2MF$ , since  $X_{(1,2)}$  is at most  $RF$  bits by definition of achievability, and since  $Z_1, Z_2$  contain each  $MF$  bits. On the other hand, since  $W_1$  can be recovered from  $(X_{(1,2)}, Z_1)$  and  $W_2$  can be recovered from  $(X_{(1,2)}, Z_2)$ , the number of bits that need to be transmitted over this cut is at least  $2F$ . Hence,

$$RF + 2MF \geq 2F,$$

so that

$$R \geq 2 - 2M.$$

As this holds for all achievable memory-rate pairs  $(M, R)$ , we conclude that

$$R^*(M) \geq 2 - 2M.$$



The second cut separates  $(X_{(1,2)}, X_{(2,1)}, Z_1)$  from  $(\hat{W}_1, \hat{W}_2)$ . This cut yields

$$2RF + MF \geq 2F$$

for any achievable memory-rate pair  $(M, R)$ , implying that

$$R^*(M) \geq 1 - M/2.$$

Together, this yields the dashed red curve in Fig. 3, which coincides with the lower bound stated in Theorem 2.

For the case  $N = K = 2$ , the memory-rate tradeoff can in fact be found exactly and is indicated by the dotted black curve in Fig. 3. This is argued by showing that the pair  $(M, R) = (1/2, 1)$  is also achievable, and by deriving the additional non cut-set bound

$$R^*(M) \geq 3/2 - 3M/2.$$

This shows that, while the bounds in Theorems 1 and 2 are sufficient to characterize the memory-rate tradeoff  $R^*(M)$  to within a bounded multiplicative gap, neither the achievable region nor the converse are tight in general. The details of this derivation are reported in the Appendix.  $\diamond$

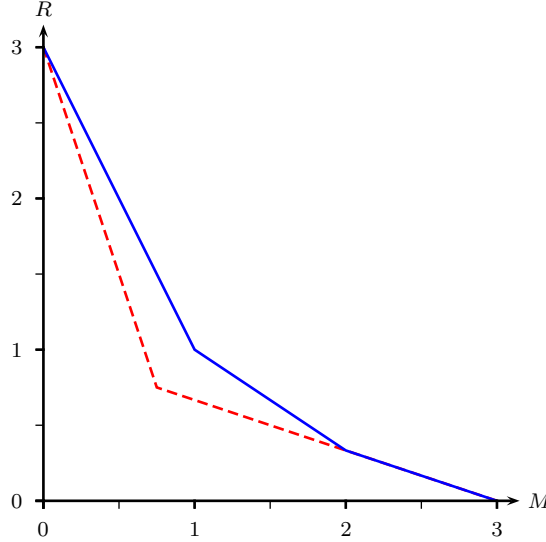


Fig. 5. Memory-rate tradeoff for  $N = 3$  files and  $K = 3$  users. The achievable rate  $R(M)$  from Theorem 1 is indicated by the solid blue curve. The lower bound on  $R^*(M)$  from Theorem 2 is indicated by the dashed red curve.

**Example 5.** In this example, we assume that  $N = K = 3$  so that there are three users and three files, say  $W_1 = A$ ,  $W_2 = B$ , and  $W_3 = C$ . Again, it is trivial to see that the  $(M, R)$  pairs  $(0, 3)$  and  $(3, 0)$  are achievable. We focus on two nontrivial cases  $M = 1$  and  $M = 2$ .

Consider first caches of size  $M = 1$ . We split each file into three subfiles of equal size, i.e.,  $A = (A_1, A_2, A_3)$ ,  $B = (B_1, B_2, B_3)$ , and  $C = (C_1, C_2, C_3)$ . In the placement phase, the cache content of user  $k$  is selected as  $Z_k = (A_k, B_k, C_k)$ . A more formal way to describe this cache placement, which is a bit exaggerated for this simple setting but will be useful for the general setting below, is as follows. Let  $\mathcal{T}$  be a subset of one element of  $\{1, 2, 3\}$ . Then subfiles  $A_{\mathcal{T}}$ ,  $B_{\mathcal{T}}$ ,  $C_{\mathcal{T}}$  are placed into the cache of user  $k$  if  $k \in \mathcal{T}$ . For example,  $A_1$  is cached at user one since in this case  $\mathcal{T} = \{1\}$ .

For the delivery phase, let us consider as an example that user one requests file  $A$ , user two requests file  $B$ , and user three requests file  $C$ . Then the missing subfiles are  $A_2$  and  $A_3$  for user one,  $B_1$  and  $B_3$  for user two, and  $C_1$  and  $C_2$  for user three. Given the cache contents, users one and two aim to exchange  $A_2$  and  $B_1$ , users one and three aim to exchange  $A_3$  and  $C_1$ , and users two and three aim to exchange  $B_3$  and  $C_2$ . The signal  $X_{(1,2,3)} = (A_2 \oplus B_1, A_3 \oplus C_1, B_3 \oplus C_2)$  enables all of these three exchanges. All

other requests can be satisfied in a similar manner. Since each subfile has rate  $1/3$ , the proposed scheme achieves a rate of 1, and therefore  $(M, R)$  pair  $(1, 1)$  is achievable.

Consider next caches of size  $M = 2$ . In this case, each file is again split into three subfiles of equal size. However, it will be convenient to label these subfiles differently, namely  $A = (A_{12}, A_{13}, A_{23})$ ,  $B = (B_{12}, B_{13}, B_{23})$ , and  $C = (C_{12}, C_{13}, C_{23})$ . In the placement phase, the caching strategy is

$$\begin{aligned} Z_1 &= (A_{12}, A_{13}, B_{12}, B_{13}, C_{12}, C_{13}), \\ Z_2 &= (A_{12}, A_{23}, B_{12}, B_{23}, C_{12}, C_{23}), \\ Z_3 &= (A_{13}, A_{23}, B_{13}, B_{23}, C_{13}, C_{23}). \end{aligned}$$

This cache placement can be understood as follows. Let  $\mathcal{T}$  be a subset of two elements of  $\{1, 2, 3\}$ . Then subfiles  $A_{\mathcal{T}}$ ,  $B_{\mathcal{T}}$ ,  $C_{\mathcal{T}}$  are placed into the cache of user  $k$  if  $k \in \mathcal{T}$ . For example,  $A_{13}$  is cached at users one and three since in this case  $\mathcal{T} = \{1, 3\}$ .

For the delivery phase, let us again assume as an example that user one requests file  $A$ , user two requests file  $B$ , and user three requests file  $C$ . In this case, user one misses subfile  $A_{23}$ , which is available at both users two and three. User two misses subfile  $B_{13}$ , which is available at both users one and three. And user three misses subfile  $C_{12}$ , which is available at both users one and two. In other words, the three users would like to exchange the subfiles  $A_{23}, B_{13}, C_{12}$ . This exchange can be enabled by transmitting the signal  $X_{(1,2,3)} = A_{23} \oplus B_{13} \oplus C_{12}$  over the shared link. Given its cache content, each user can then recover the missing subfile. All other requests can be satisfied in a similar manner. The rate of transmission in the delivery phase is  $1/3$ , and therefore  $(M, R)$  pair  $(2, 1/3)$  is achievable.

The arguments so far show that the solid blue curve in Fig. 5 is achievable. This coincides with the upper bound in Theorem 1.

For the lower bound on  $R^*(M)$ , we use two cut-set bounds. The first cut separates  $(X_{(1,2,3)}, Z_1, Z_2, Z_3)$  from  $(\hat{A}, \hat{B}, \hat{C})$ . This is a valid cut since  $(X_{(1,2,3)}, Z_1)$ ,  $(X_{(1,2,3)}, Z_2)$ , and  $(X_{(1,2,3)}, Z_3)$  determine  $A$ ,  $B$ , and  $C$ , respectively. For any achievable  $(M, R)$  pair, the capacity of this cut is at most  $RF + 3MF$ , and the number of bits that need to be transmitted over it is at least  $3F$ . Hence,

$$RF + 3MF \geq 3F,$$

which implies that

$$R^*(M) \geq 3 - 3M.$$

The second cut separates  $(X_{(1,2,3)}, X_{(3,1,2)}, X_{(2,3,1)}, Z_1)$  from  $(\hat{A}, \hat{B}, \hat{C})$ . This is a valid cut since  $(X_{(1,2,3)}, Z_1)$ ,  $(X_{(3,1,2)}, Z_1)$ , and  $(X_{(2,3,1)}, Z_1)$  determine  $A$ ,  $C$ , and  $B$ , respectively. For any achievable  $(M, R)$  pair, the cut capacity is at most  $3RF + MF$ , and the number of bits that need to be transmitted over it is at least  $3F$ . Hence,

$$3RF + MF \geq 3F,$$

which implies that

$$R^*(M) \geq 1 - M/3.$$

Together, these two cut-set bounds result in the dashed red curve in Fig. 5. This coincides with the lower bound in Theorem 2.  $\diamond$

## V. AN ORDER-OPTIMAL CACHING STRATEGY (PROOF OF THEOREM 1)

We now present the general achievable scheme. To simplify the description of the algorithm, the notation

$$\begin{aligned} [N] &\triangleq \{1, \dots, N\}, \\ [K] &\triangleq \{1, \dots, K\} \end{aligned}$$

is used. We first describe the algorithm in words. Consider cache size  $M \in \frac{N}{K} \cdot \{0, 1, \dots, K\}$ , and set

$$t \triangleq MK/N.$$

Observe that  $t$  is an integer between 0 and  $K$ .

If  $M = 0$ , then in the delivery phase the server can simply transmit the union of all requested files over the shared link, resulting in  $F \min\{N, K\}$  bits to be sent. Hence

$$R^*(0) \leq \min\{N, K\}.$$

If  $M = N$ , then all files in the database can be cached at every user in the placement phase. Hence

$$R^*(N) = 0.$$

Assume in the following that  $M$  is strictly larger than zero and strictly less than  $N$ , so that  $t \in \{1, 2, \dots, K-1\}$ .

In the placement phase, each file is split into  $\binom{K}{t}$  nonoverlapping subfiles of equal size. It will be convenient to label the subfiles of file  $W_n$  as

$$W_n = (W_{n,\mathcal{T}} : \mathcal{T} \subset [K], |\mathcal{T}| = t).$$

For each  $n \in [N]$ , subfile  $W_{n,\mathcal{T}}$  is placed in the cache of user  $k$  if  $k \in \mathcal{T}$ . Thus, each user caches a total of  $N \binom{K-1}{t-1}$  subfiles. Since each of these subfiles has size  $F/\binom{K}{t}$ , this requires

$$N \binom{K-1}{t-1} \frac{F}{\binom{K}{t}} = F \frac{Nt}{K} = FM$$

bits of cache memory at each user, satisfying the memory constraint.

**Example 6.** Let  $N = K = 3$  and  $M = 2$ . Then  $t = 2$  and the cache placement is

$$\begin{aligned} Z_1 &= (W_{n,\{1,2\}}, W_{n,\{1,3\}})_{n=1}^3, \\ Z_2 &= (W_{n,\{1,2\}}, W_{n,\{2,3\}})_{n=1}^3, \\ Z_3 &= (W_{n,\{1,3\}}, W_{n,\{2,3\}})_{n=1}^3, \end{aligned}$$

as we have already seen in Example 5 in Section IV.  $\diamond$

We next describe the delivery phase. Consider the request vector  $(d_1, \dots, d_K) \in [N]^K$ , i.e., user  $k$  requests file  $W_{d_k}$ . We focus on a subset  $\mathcal{S} \subset [K]$  of  $|\mathcal{S}| = t + 1$  users. Observe that every  $t$  users in  $\mathcal{S}$  share a subfile in their caches that is needed at the remaining user in  $\mathcal{S}$ . More precisely, fix a user  $k \in \mathcal{S}$ , and note that  $|\mathcal{S} \setminus \{k\}| = t$ . The subfile  $W_{d_k, \mathcal{S} \setminus \{k\}}$  is requested by user  $k$  since it is a subfile of  $W_{d_k}$ . At the same time, it is missing at user  $k$  since  $k \notin \mathcal{S} \setminus \{k\}$ . Finally, it is present in the cache of any user  $s \in \mathcal{S} \setminus \{k\}$ .

For each such subset  $\mathcal{S} \subset [K]$  of cardinality  $|\mathcal{S}| = t + 1$ , the server transmits

$$\bigoplus_{s \in \mathcal{S}} W_{d_s, \mathcal{S} \setminus \{s\}},$$

where  $\oplus$  denotes bitwise XOR. Each of these sums results in  $F/\binom{K}{t}$  bits being sent over the shared link. Since the number of subsets  $\mathcal{S}$  is  $\binom{K}{t+1}$ , the total number of bits sent over the shared link is

$$\begin{aligned} FR &= \binom{K}{t+1} \frac{F}{\binom{K}{t}} \\ &= F \frac{K-t}{t+1} \\ &= F \frac{K(1-M/N)}{1+KM/N}, \end{aligned}$$

where we have used that  $t = MK/N$ .

**Example 7.** Let  $N = K = 3$  and  $M = 1$ . Then  $t = 1$  and the cache placement is  $Z_k = (W_{n,\{k\}})_{n=1}^3$ . For request  $(d_1, d_2, d_3) = (1, 2, 3)$ , the signal transmitted in the delivery phase is

$$X_{(1,2,3)} = (W_{1,\{2\}} \oplus W_{2,\{1\}}, W_{1,\{3\}} \oplus W_{3,\{1\}}, W_{2,\{3\}} \oplus W_{3,\{2\}}),$$

as we have already seen in Example 5 in Section IV. Here, the three sums correspond to  $\mathcal{S} = \{1, 2\}$ ,  $\mathcal{S} = \{3, 1\}$ , and  $\mathcal{S} = \{3, 2\}$ , respectively.  $\diamond$

We now argue that each user can successfully recover its requested message. Consider again a subset  $\mathcal{S} \subset [K]$  with  $|\mathcal{S}| = t + 1$ . Since each user  $k \in \mathcal{S}$  already has access to the subfiles  $W_{d_s, \mathcal{S} \setminus \{k\}}$  for all  $s \in \mathcal{S} \setminus \{k\}$ , it can solve for  $W_{d_k, \mathcal{S} \setminus \{k\}}$  from the signal

$$\bigoplus_{s \in \mathcal{S}} W_{d_s, \mathcal{S} \setminus \{s\}}$$

sent over the shared link. Since this is true for every such subset  $\mathcal{S}$ , each receiver  $k$  is able to recover all subfiles of the form

$$\{W_{d_k, \mathcal{T}} : \mathcal{T} \subset [K] \setminus \{k\}, |\mathcal{T}| = t\}$$

of the requested file  $W_{d_k}$ . The remaining subfiles are of the form  $W_{d_k, \mathcal{T}}$  for  $\mathcal{T}$  such that  $k \in \mathcal{T}$ . But these subfiles are already available in the cache of user  $k$ . Hence each user  $k$  can recover all subfiles of its requested file  $W_{d_k}$ .

This shows that, for  $M \in \frac{N}{K} \cdot \{1, 2, \dots, K - 1\}$ ,

$$R^*(M) \leq \frac{K(1 - M/N)}{1 + KM/N}$$

As was pointed out earlier (see Example 4 in Section IV), the points on the line connecting any two achievable points are also achievable. Together with the upper bound for  $M \in \{0, N\}$  derived earlier, this proves Theorem 1.A. Similarly, taking the minimum between the rate derived here and the rate

$$\min\{N, K\}(1 - M/N)$$

achieved by the conventional scheme described in Example 1 in Section II-A proves Theorem 1.B.  $\blacksquare$

For completeness, we now formally describe the caching algorithm for  $N$  files,  $K$  users, and cache size  $M$  such that  $MK/N$  is a positive integer less than to  $K$ .

**procedure** PLACEMENT( $W_1, \dots, W_N$ )

$t \leftarrow MK/N$

$\mathfrak{T} \leftarrow \{\mathcal{T} \subset [K] : |\mathcal{T}| = t\}$

**for**  $n \in [N]$  **do**

split  $W_n$  into  $(W_{n, \mathcal{T}} : \mathcal{T} \in \mathfrak{T})$  of equal size

**end for**

**for**  $k \in [K]$  **do**

$Z_k \leftarrow (W_{n, \mathcal{T}} : n \in [N], \mathcal{T} \in \mathfrak{T}, k \in \mathcal{T})$

**end for**

**end procedure**

**procedure** DELIVERY( $W_1, \dots, W_N, d_1, \dots, d_K$ )

$t \leftarrow MK/N$

$\mathfrak{S} \leftarrow \{\mathcal{S} \subset [K] : |\mathcal{S}| = t + 1\}$

$X_{(d_1, \dots, d_K)} \leftarrow (\bigoplus_{k \in \mathcal{S}} W_{d_k, \mathcal{S} \setminus \{k\}} : \mathcal{S} \in \mathfrak{S})$

**end procedure**

## VI. CONVERSE (PROOF OF THEOREM 2)

Let  $s \in \{1, \dots, \min\{N, K\}\}$  and consider the first  $s$  caches  $Z_1, \dots, Z_s$ . There exists an input to the shared link, say  $X_1$ , such that  $X_1$  and  $Z_1, \dots, Z_s$  determine the files  $W_1, \dots, W_s$ . Similarly, there exists an input to the shared link, say  $X_2$ , such that  $X_2$  and  $Z_1, \dots, Z_s$  determine the files  $W_{s+1}, \dots, W_{2s}$ . Continue in the same manner selecting appropriate  $X_3, \dots, X_{\lfloor N/s \rfloor}$ , see Fig. 6. We then have that  $X_1, \dots, X_{\lfloor N/s \rfloor}$  and  $Z_1, \dots, Z_s$  determine  $W_1, \dots, W_{s\lfloor N/s \rfloor}$ .

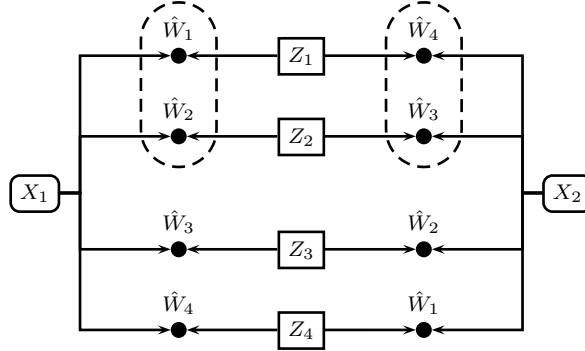


Fig. 6. Cut corresponding to parameter  $s = 2$  in the proof of the converse. In the figure,  $N = K = 4$ , and  $X_1 = X_{(1,2,3,4)}$ ,  $X_2 = X_{(4,3,2,1)}$ .

Consider then the cut separating  $X_1, \dots, X_{\lfloor N/s \rfloor}$  and  $Z_1, \dots, Z_s$  from the corresponding receivers as indicated in Fig. 6. By the cut-set bound [10, Theorem 14.10.1],

$$s\lfloor N/s \rfloor \leq \lfloor N/s \rfloor R^*(M) + sM.$$

Solving for  $R^*$  and optimizing over all possible choices of  $s$ , we obtain

$$R^*(M) \geq \max_{s \in \{1, \dots, \min\{N, K\}\}} \left( s - \frac{s}{\lfloor N/s \rfloor} M \right),$$

proving the theorem. ■

## VII. APPROXIMATION OF $R^*(M)$ (PROOF OF THEOREM 3)

We now compare the upper bound on  $R^*(M)$  in Theorem 2 to the rate  $R(M)$  achieved by the proposed scheme given by the lower convex envelope of the points

$$R(M) = \min \left\{ \frac{K(1 - M/N)}{1 + KM/N}, N - M \right\}$$

for  $M$  a multiple of  $N/K$  as described in Theorem 1. The proof is based on the following observation. The function  $R(M)$  has three distinct regimes as depicted in Fig. 7. The first regime is for  $M$  between 0 and approximately  $\max\{2, N/K\}$ . In this regime,  $R(M)$  is close to linear. The second regime is for  $M$  approximately between  $\max\{2, N/K\}$  and  $N/2$ , in which  $R(M)$  is nonlinear and behaves essentially like  $N/M$ . The third regime is for  $M$  between approximately  $N/2$  and  $N$ , in which  $R(M)$  is again close to linear. We bound the ratio  $R^*(M)/R(M)$  separately in each of these regimes—though, to optimize the constants, we will choose slightly different definitions of the boundaries for the three regions.

It will be convenient to consider the two cases  $\min\{N, K\} \leq 12$  and  $\min\{N, K\} \geq 13$  separately. We have

$$R(M) \leq \min\{N, K\}(1 - M/N)$$

by Theorem 1. On the other hand, setting  $s = 1$  in Theorem 2 yields

$$R^*(M) \geq 1 - M/N.$$

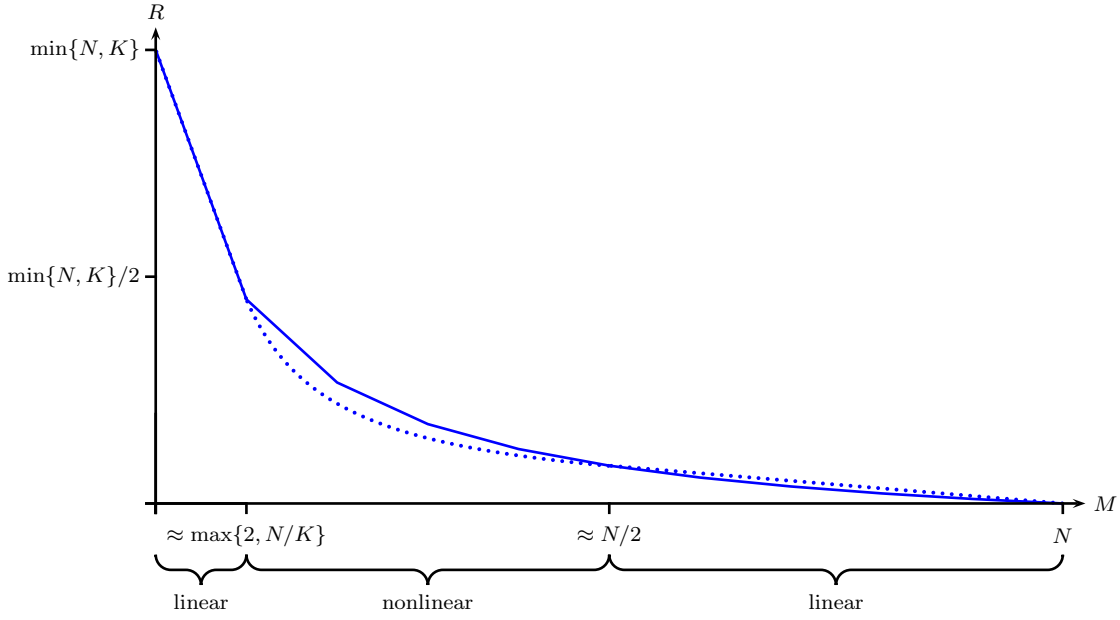


Fig. 7. Achievable rate  $R(M)$  (solid curve) as defined in Theorem 1 for  $N = 20$  files and  $K = 10$ . The function  $R(M)$  has three distinct regimes, which are approximately  $0 \leq M \leq \max\{2, N/K\}$ ,  $\max\{2, N/K\} < M \leq N/2$ , and  $N/2 < M \leq N$ . In the first and third regimes,  $R(M)$  is essentially linear; in the second regime,  $R(M)$  is nonlinear (as indicated by the dotted curve).

Hence,

$$\frac{R(M)}{R^*(M)} \leq \min\{N, K\} \leq 12 \quad (2)$$

for  $\min\{N, K\} \leq 12$ .

Assume in the following that  $\min\{N, K\} \geq 13$ . We consider the three cases  $0 \leq M \leq 1.1 \max\{1, N/K\}$ ,  $1.1 \max\{1, N/K\} < M \leq 0.092N$ , and  $0.092N < M \leq N$  separately. Assume first that  $0 \leq M \leq 1.1 \max\{1, N/K\}$ . We have

$$R(M) \leq R(0) \leq \min\{N, K\} \quad (3)$$

by Theorem 1. By Theorem 2, and using that  $\lfloor N/s \rfloor \geq N/s - 1$ ,

$$R^*(M) \geq s - \frac{s^2}{1 - s/N} \frac{M}{N}.$$

Setting  $s = \lfloor 0.275 \min\{N, K\} \rfloor \in \{1, \dots, \min\{N, K\}\}$ , we obtain for  $M \leq 1.1 \max\{1, N/K\}$

$$\begin{aligned} R^*(M) &\geq R^*(1.1 \max\{1, N/K\}) \\ &\geq \lfloor 0.275 \min\{N, K\} \rfloor - \frac{(\lfloor 0.275 \min\{N, K\} \rfloor)^2}{1 - \lfloor 0.275 \min\{N, K\} \rfloor / N} \frac{1.1 \max\{1, N/K\}}{N} \\ &\geq \min\{N, K\} \left( 0.275 - \frac{1}{\min\{N, K\}} - \frac{1.1 \cdot 0.275^2}{1 - 0.275 \min\{1, K/N\}} \right) \\ &\geq \min\{N, K\} \left( 0.275 - \frac{1}{13} - \frac{1.1 \cdot 0.275^2}{1 - 0.275} \right) \\ &\geq \frac{1}{12} \min\{N, K\}, \end{aligned} \quad (4)$$

where we have used the assumption that  $\min\{N, K\} \geq 13$ . Combining (3) and (4) yields

$$\frac{R(M)}{R^*(M)} \leq 12 \quad (5)$$

for  $0 \leq M \leq 1.1 \max\{1, N/K\}$  and  $\min\{N, K\} \geq 13$ .

Assume then that  $1.1 \max\{1, N/K\} < M \leq 0.092N$ . Let  $\tilde{M}$  be the largest multiple of  $N/K$  less than or equal to  $M$ , so that

$$0 \leq M - N/K \leq \tilde{M} \leq M.$$

Then, by Theorem 1,

$$\begin{aligned} R(M) &\leq R(\tilde{M}) \\ &\leq \frac{K(1 - \tilde{M}/N)}{1 + K\tilde{M}/N} \\ &\leq \frac{K(1 - M/N + 1/K)}{1 + KM/N - 1} \\ &\leq N/M, \end{aligned} \tag{6}$$

where we have used that  $M/N \geq 1/K$  in the last inequality. Setting  $s = \lfloor 0.3N/M \rfloor \in \{1, \dots, \min\{N, K\}\}$  in Theorem 2, we obtain

$$\begin{aligned} R^*(M) &\geq s - \frac{s^2}{N - s}M \\ &\geq \frac{0.3N}{M} - 1 - \frac{0.3^2 N^2 / M^2}{N - 0.3N/M}M \\ &\geq \frac{N}{M} \left( 0.3 - 0.092 - \frac{0.3^2}{1 - 0.3/1.1} \right) \\ &\geq \frac{N}{12M}. \end{aligned} \tag{7}$$

Combining (6) and (7) yields

$$\frac{R(M)}{R^*(M)} \leq 12 \tag{8}$$

for  $1.1 \max\{1, N/K\} < M \leq 0.092N$ .

Finally, assume  $0.092N < M \leq N$ . Let  $\tilde{M}$  be the largest multiple of  $N/K$  less than or equal to  $0.092N$ , so that

$$0 \leq 0.092N - N/K \leq \tilde{M} \leq 0.092N.$$

Then, using convexity of  $R(\cdot)$  and that  $R(N) = 0$ ,

$$\begin{aligned} R(M) &\leq \frac{R(\tilde{M})}{1 - \tilde{M}/N} (1 - M/N) \\ &\leq \frac{1}{1 - 0.092} R(\tilde{M}) (1 - M/N), \end{aligned}$$

where we have used that  $\tilde{M} \leq 0.092N$ . Now, by Theorem 1,

$$\begin{aligned} R(\tilde{M}) &\leq \frac{1}{\tilde{M}/N + 1/K} \\ &\leq \frac{1}{0.092 - 1/K + 1/K} \\ &= \frac{1}{0.092}. \end{aligned}$$

Hence,

$$\begin{aligned} R(M) &\leq \frac{1}{0.092(1 - 0.092)}(1 - M/N) \\ &\leq 12(1 - M/N). \end{aligned} \quad (9)$$

Setting  $s = 1$  in Theorem 2, we obtain

$$R^*(M) \geq 1 - M/N. \quad (10)$$

Combining (9) and (10) yields

$$\frac{R(M)}{R^*(M)} \leq 12 \quad (11)$$

for  $0.092N < M \leq N$  and  $\min\{N, K\} \geq 13$ .

Combining (2), (5), (8), and (11) yields

$$\frac{R(M)}{R^*(M)} \leq 12$$

for all  $N, K$  and all  $0 \leq M \leq N$ . ■

## APPENDIX

Example 4 in Section IV derives the upper and lower bounds on the memory-rate tradeoff  $R^*(M)$  claimed in Theorems 1 and 2 for  $K = N = 2$ . While these two bounds are sufficient to characterize  $R^*(M)$  to within a bounded multiplicative gap, as we show in what follows, neither the achievable region nor the converse are tight in general.

We start by arguing that the achievable scheme can be improved by focusing on the case  $M = 1/2$ . As before, we split each file into two subfiles, i.e.,  $A = (A_1, A_2)$  and  $B = (B_1, B_2)$ . In the placement phase, we choose the cache contents as  $Z_1 = A_1 \oplus B_1$  and  $Z_2 = A_2 \oplus B_2$ . Assume that user one requests file  $A$  and user two requests file  $B$ . The server can satisfy these requests by transmitting  $X_{(1,2)} = (B_1, A_2)$  at rate  $R = 1$ . The other three possible requests can be satisfied in a similar manner. This shows that  $(M, R)$  pair  $(1/2, 1)$  is achievable. This new point improves the boundary of the achievable region from the solid blue to the dotted black curve in Fig. 3 in Section IV.

We now argue that the converse can also be improved. Referring to Fig. 2(b) in Section II, we have for any achievable memory-rate pair  $(M, R)$ ,

$$\begin{aligned} 2RF + 2MF &\geq H(X_{(1,2)}, Z_1) + H(X_{(2,1)}, Z_2) \\ &= H(X_{(1,2)}, Z_1|W_1) + H(X_{(2,1)}, Z_2|W_1) + I(W_1; X_{(1,2)}, Z_1) + I(W_1; X_{(2,1)}, Z_2) \\ &\geq H(X_{(1,2)}, Z_1, X_{(2,1)}, Z_2|W_1) + I(W_1; X_{(1,2)}, Z_1) + I(W_1; X_{(2,1)}, Z_2) \\ &\geq I(W_2; X_{(1,2)}, Z_1, X_{(2,1)}, Z_2|W_1) + I(W_1; X_{(1,2)}, Z_1) + I(W_1; X_{(2,1)}, Z_2). \end{aligned}$$

Now, since  $W_1$  can be decoded from  $(X_{(1,2)}, Z_1)$  and also from  $(X_{(2,1)}, Z_2)$ , and since  $W_2$  can be decoded from  $(X_{(1,2)}, Z_1, X_{(2,1)}, Z_2)$ , we obtain using Fano's inequality that

$$\begin{aligned} I(W_1; X_{(1,2)}, Z_1) &\geq F - \varepsilon F, \\ I(W_1; X_{(2,1)}, Z_2) &\geq F - \varepsilon F, \\ I(W_2; X_{(1,2)}, Z_1, X_{(2,1)}, Z_2|W_1) &\geq F - \varepsilon F \end{aligned}$$

for any  $\varepsilon > 0$  as long as  $F$  is large enough. Hence,

$$2RF + 2MF \geq 3F - 3\varepsilon F.$$



Since  $\varepsilon > 0$  is arbitrary, this shows that

$$R \geq 3/2 - 3M/2.$$

Since this is true for any achievable  $(M, R)$  pair, this implies that

$$R^*(M) \geq 3/2 - 3M/2.$$

This bound, together with the two cut-set bounds derived earlier, proves that the dotted black curve depicted in Fig. 3 in Section IV is indeed equal to  $R^*(M)$ .

## REFERENCES

- [1] L. W. Dowdy and D. V. Foster, "Comparative models of the file assignment problem," *ACM Comput. Surv.*, vol. 14, pp. 287–313, June 1982.
- [2] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *IEEE J. Sel. Areas Commun.*, vol. 14, pp. 1110–1122, Aug. 1996.
- [3] A. Dan, D. Sitaram, and P. Shahabuddin, "Dynamic batching policies for an on-demand video server," *Multimedia Syst.*, vol. 4, pp. 112–121, June 1996.
- [4] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, "Placement algorithms for hierarchical cooperative caching," in *Proc. ACM-SIAM SODA*, pp. 586–595, Jan. 1999.
- [5] A. Meyerson, K. Munagala, and S. Plotkin, "Web caching using access statistics," in *Proc. ACM-SIAM SODA*, pp. 354–363, 2001.
- [6] I. Baev, R. Rajaraman, and C. Swamy, "Approximation algorithms for data placement problems," *SIAM J. Comput.*, vol. 38, pp. 1411–1429, July 2008.
- [7] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, pp. 1–9, Mar. 2010.
- [8] Y. Birk and T. Kol, "Informed-source coding-on-demand (ISCOD) over broadcast channel," in *Proc. IEEE INFOCOM*, pp. 1257–1264, Mar. 1998.
- [9] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol, "Index coding with side information," in *Proc. IEEE FOCS*, pp. 197–206, Oct. 2006.
- [10] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley, 1991.